ITS Technical Bulletin 219
NEW IBM HIGH LEVEL ASSEMBLER

Issued Date:	28 Aug 1995
Effective Date:	30 Aug 1995
Section/Groups:	Security/Software Support
Submitted By:	Farrell Wiser
Approved By:	Clair Christensen

IBM's new assembler, named 'HIGH LEVEL ASSEMBLER', has been installed on CPU0 and CPU1 for testing. It resides in 'SYS1.LINKLIB' and 'SYS1.PROCLIB'. Samples are in 'SYS1.SAMPLIB'. It is generally 'downward compatible' (except as noted below), so your old processes should work unchanged. PLEASE USE THIS OPPORTUNITY TO TEST IT, as it will be migrated to the 'Application Testing'' environment (CPU4) on Oct 2, 1995.

The current plan is to install 'High Level Assembler' on CPU4 on Oct. 2, 1995, for 'Production Testing'. When sufficient testing has been done to satisfy reliability and migration requirements, it will be installed on CPU2 and CPU3.

If you do not have direct access to CPU0 or CPU1 but wish to test on CPU0 or CPU1 prior to October 2, 1995, use PROC ASMHC0 instead of ASMHC, ASMHCG0 instead of ASMHCG, ASMHCL0 instead of ASMHCL, and ASMHCLG0 instead of ASMHCLG. These PROCS will be available until October 2, 1995.

Migration Considerations

(The following is Copyright IBM Corp. 1982, 1995, information.)

Source Programs: Migration from High Level Assembler Release 1, or Assembler H Version 2, to High Level Assembler Release 2 requires an analysis of existing assembler language programs to ensure that they do not contain macro instructions with names that conflict with the High Level Assembler Release 2 symbolic operation codes, or SET symbols with names that conflict with the names of High Level Assembler Release 2 system variable symbols.

With the exception of these possible conflicts, and with appropriate High Level Assembler option values, assembler language source programs written for High Level Assembler Release 1, or Assembler H Version 2, that assemble without warning or error diagnostic messages, should assemble correctly using High Level Assembler Release 2.

Object Programs: Object programs generated by High Level Assembler Release 2 in any one of the supported operating systems can be migrated to any other of the supported operating systems for execution. The object programs being migrated must be link-edited in the target operating system environment before execution.

You should be aware of the differences in the code generated by system macros in the supported operating systems. Operational facilities available on the source operating system but not available on the target operating system should not be specified for any program which is required to be compatible, either during assembly or link-edit.

Information about the new HLASM, and changes/mods/errata.

Highlights of High Level Assembler

(The following is Copyright IBM Corp. 1981, 1995, information.)

High Level Assembler is a functional replacement for Assembler H Version 2. It offers all the proven facilities provided by these earlier assemblers, and many new facilities designed to improve programmer productivity and simplify assembler language program development and maintenance.

Some of the highlights of High Level Assembler are:

Extensions to the basic assembler language.

Extensions to the macro and conditional assembly language, including external function calls and built-in functions.

Enhancements to the assembly listing, including a new macro and copy code member cross reference section, and a new section that lists all the unreferenced symbols defined in CSECTS.

New assembler options.

A new associated data file, the ADATA file, containing both language-dependent and language-independent records that can be used by debugging and other tools.

A DOS operation code table to assist in migration from DOS/VSE Assembler.

The use of 31-bit addressing for most working storage requirements.

An extended object format data set.

Internal performance enhancements and diagnostic capabilities.

Performance and Usability

(The following is Copyright IBM Corp. 1981, 1995, information.)

The following enhancements improve system performance and system usability:

Virtual Storage Constraint Relief: The High Level Assembler modules and data areas, with some exceptions, use 31-bit addressing.

Extended Object Format Support: The XOBJECT assembler option instructs the High Level Assembler to produce an extended object format data set. Refer to DFSMS/MVS Version 1 Release 3 Program Management, SC26-4916 for more information.

Associated Data Architecture: The structure and content of the ADATA records has been improved. A new ADATA instruction lets you generate user records during assembly of the source program.

User Exits: The EXIT assembler option now supports a user-supplied terminal I/O routine to be used in place of, or in addition to, the terminal processing of the High Level Assembler.

Sample Exits: High Level Assembler provides sample I/O exits to complement the assembler's output processing for the following:

- ADATA This exit provides you with an interface to the assembler that lets you write your own filter routines to examine and extract information from ADATA records.

- LISTING This exit lets you selectively print parts of the assembler listing.

- SOURCE This exit lets you read source statements from variable-length input data sets.

System-Determined Blocksize: The High Level Assembler supports the system-determined blocksize (SDB) feature of MVS/DFP* in the MVS/ESA environment. SDB allows the blocksize for all output datasets, except SYSLIN and SYSPUNCH, to be set to the optimum, system-determined value.

Programmer Productivity

(The following is Copyright IBM Corp. 1981, 1995, information.)

The following enhancements simplify program development and increase programmer productivity:

Source Program Assembler Options: You can use the new *PROCESS statement to specify selected assembler options in the assembler source program.

Profile Option: The PROFILE assembler option lets you specify the name of a library member containing assembler source statements which the assembler inserts at the start of the assembler source program. The statements in the member are processed after any ICTL instruction or *PROCESS statements.

Translating Character Constants and Literals: The TRANSLATE assembler option lets you specify a translation table that the assembler uses to convert characters contained in character (C-type) data constants (DCs) and literals. You can use the ASCII translation table provided with High Level Assembler or use your own translation table.

Record Numbers: High Level Assembler now provides an absolute record number, and a relative record number within each data set, for each record that is read from an input data set, or written to an output data set. These record numbers are passed to any user exit you specify for the assembly. The relative record number is used for information messages produced when the new FLAG(RECORD) assembler option is specified, and it is also written to the associated data file in the source analysis record.

Built-In Functions for Conditional Assembly Instructions: High Level Assembler provides new built-in functions for conditional assembly instructions that perform logical, arithmetic, and character string operations in SETA, SETB and SETC expressions.

External Function Calls: The new SETAF and SETCF instructions let you supply your own routines to perform external functions for conditional assembly, and place the results in a variable (SET) symbol. You may write these routines in any programming language that conforms to standard OS Linkage conventions.

System Variable Symbols: There is a new system variable for each data set name, volume serial number, and member name of all High Level Assembler output data sets.

National Language Support

(The following is Copyright IBM Corp. 1981, 1995, information.)

The LANGUAGE assembler option lets you select the language that the assembler uses to produce diagnostic messages. The assembler provides diagnostic messages in the following languages:

English mixed case
English uppercase
German
Japanese
Spanish

NOTE:
The assembler uses the language specified in the installation default options for messages produced in CMS by the ASMAHL command.

Diagnostic Information

(The following is Copyright IBM Corp. 1981, 1995, information.)

Some of the new diagnostic and listing enhancements are:

Enhanced statement continuation checking

Informational messages to help you locate the original source statement that generated diagnostic messages

Assembler listing headings printed in either uppercase or a mixture of uppercase and lowercase alphabetic characters

An Unreferenced Symbols Defined in CSECTs section of the assembler listing

An Extended Source and Object section of the assembler listing to support 31-bit addresses

Eight-digit address and length fields to support 31-bit addresses

Six-digit statement numbers with leading zeroes suppressed

A Macro and Copy Code Cross Reference section of the assembler listing

A new compact format of the Ordinary Symbol and Literal Cross Reference section of the assembler listing

Improved page break handling in conjunction with the EJECT, SPACE, and TITLE assembler instructions

This is documented in High Level Assembler for MVS Programmers Guide, IBM Form # SC26-4941-01. Online Bookmanager Book name = ASMP1001. See also, High Level Assembler Language Reference, IBM Form # SC26-4940-01. Online Bookmanager Book name = ASMR1001. The online books are part of BookShelf name = ASMSH001.

Additional Information

The IBM-Supplied ‘default’ OPTIONS are:

ADATA=NOADATA   ALIGN=ALIGN   ASA=NOASA
BATCH=BATCH   COMPAT=NOCOMPAT   DBCS=NODBCS
DECK=NODECK   DISK=(see PRINT)   DXREF=DXREF   ERASE=ERASE
ESD=ESD   EXIT=NOEXIT FLAG=(0,ALIGN,CONT,RECORD,NOSUBSTR)
FOLD=NOFOLD   LANGUAGE=EN   LIBMAC=NOLIBMAC   LINECOUNT=60

```
          LIST=121       MXREF=SOURCE    NOSEG=(none)          OBJECT=OBJECT
OPTABLE=UNI       PCONTROL=NOPCONTROL PROFILE=NOPROFILE     PRINT=DISK
RA2=NORA2            RENT=NORENT       RLD=RLD       SIZE=MAX   SYSPARM=(null)
          TERM=NOTERM       TEST=NOTEST       TRANSLATE=NOTRANSLATE
USING=(MAP,WARN(15))XOBJECT=NOXOBJECT       XREF=(SHORT,UNREFS)
```

The new PROCs are shown below.

From SYS1.PROCLIB(ASMHC):

```
******************************** Top of Data **********************************
//ASMAC    PROC     00001000
//*       00002000
//***   ASMAC          00003000
//*       00004000
//* THIS PROCEDURE RUNS THE HIGH LEVEL ASSEMBLER AND CAN BE USED
          00005000
//* TO ASSEMBLE PROGRAMS.    00006000
//*       00007000
//ASM        EXEC PGM=ASMA90   00008000
//SYSLIB    DD  DSN=SYS1.MACLIB,DISP=SHR           00009000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,
00010000
//            DCB=BUFNO=1    00011000
//SYSPRINT DD  SYSOUT=*          00012000
//SYSLIN   DD  DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,     00013000
//            DISP=(MOD,PASS),          00014000
//            DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)   00015000
******************************* Bottom of Data *******************************
```

From SYS1.PROCLIB(ASMHCG):

```
******************************** Top of Data **********************************
//ASMACG   PROC   00001000
//*       00002000
//***   ASMACG       00003000
//*       00004000
//* THIS PROCEDURE RUNS THE HIGH LEVEL ASSEMBLER AND WILL USE
00005000
//* THE LOADER PROGRAM TO RUN THE NEWLY ASSEMBLED PROGRAM.
00006000
//*       00007000
//ASM        EXEC PGM=ASMA90   00008000
//SYSLIB    DD  DSN=SYS1.MACLIB,DISP=SHR           00009000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,
00010000
```

```
//            DCB=BUFNO=1    00011000
//SYSPRINT DD  SYSOUT=*          00012000
//SYSLIN     DD  DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,   00013000
//            DISP=(MOD,PASS),          00014000
//            DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)   00015000
//GO       EXEC PGM=LOADER,PARM='MAP,LET,PRINT',COND=(8,LT,ASM)
00016000
//SYSLIN   DD  DSN=&&OBJ,DISP=(OLD,DELETE)       00017000
//        DD  DDNAME=SYSIN         00018000
//SYSLOUT  DD  SYSOUT=*          00019000
******************************** Bottom of Data ********************************
```

From SYS1.PROCLIB(ASMHCL):

```
********************************* Top of Data **********************************
//ASMACL   PROC    00001000
//*        00002000
//***   ASMACL        00003000
//*        00004000
//* THIS PROCEDURE RUNS THE HIGH LEVEL ASSEMBLER, LINK-EDITS THE
00005000
//* NEWLY ASSEMBLED PROGRAM.      00006000
//*        00007000
//ASM        EXEC PGM=ASMA90   00008000
//SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR  00009000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,
00010000
//            DCB=BUFNO=1    00011000
//SYSPRINT DD  SYSOUT=*          00012000
//SYSLIN   DD  DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,     00013000
//            DISP=(MOD,PASS),          00014000
//            DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)   00015000
//LKED      EXEC PGM=HEWL,PARM='MAP,LET,LIST,NCAL',COND=(8,LT,ASM)
00016000
//SYSLIN   DD  DSN=&&OBJ,DISP=(OLD,DELETE)       00017000
//        DD  DDNAME=SYSIN         00018000
//SYSLMOD  DD  DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1,1)),           00019000
//            DSN=&&GOSET(GO)       00020000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(1024,(120,120),,,ROUND),UNIT=VIO,
00021000
//            DCB=BUFNO=1    00022000
//SYSPRINT DD  SYSOUT=*          00023000
******************************** Bottom of Data ********************************
```
From SYS1.PROCLIB(ASMHCLG):

```
******************************** Top of Data **********************************
//ASMACLG  PROC  00001000
//*       00002000
//***   ASMACLG     00003000
//*       00004000
//* THIS PROCEDURE RUNS THE HIGH LEVEL ASSEMBLER, LINK-EDITS THE
00005000
//* NEWLY ASSEMBLED PROGRAM AND RUNS THE PROGRAM AFTER  00006000
//* THE LINK-EDIT IS ACCOMPLISHED.            00007000
//*       00008000
//ASM       EXEC PGM=ASMA90    00009000
//SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR  00010000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(4096,(120,120),,,ROUND),UNIT=VIO,
00011000
//            DCB=BUFNO=1    00012000
//SYSPRINT DD  SYSOUT=*          00013000
//SYSLIN   DD  DSN=&&OBJ,SPACE=(3040,(40,40),,,ROUND),UNIT=VIO,     00014000
//            DISP=(MOD,PASS),          00015000
//            DCB=(BLKSIZE=3040,LRECL=80,RECFM=FBS,BUFNO=1)   00016000
//LKED     EXEC PGM=HEWL,PARM='MAP,LET,LIST',COND=(8,LT,ASM)    00017000
//SYSLIN   DD  DSN=&&OBJ,DISP=(OLD,DELETE)       00018000
//        DD  DDNAME=SYSIN         00019000
//SYSLMOD  DD  DISP=(,PASS),UNIT=SYSDA,SPACE=(CYL,(1,1,1)),           00020000
//            DSN=&&GOSET(GO)       00021000
//SYSUT1   DD  DSN=&&SYSUT1,SPACE=(1024,(120,120),,,ROUND),UNIT=VIO,
00022000
//            DCB=BUFNO=1    00023000
//SYSPRINT DD  SYSOUT=*          00024000
//GO       EXEC PGM=*.LKED.SYSLMOD,COND=((8,LT,ASM),(8,LT,LKED))
00025000
******************************* Bottom of Data *******************************
```